librice

A sans-IO ICE networking library

2024 GStreamer conference

Matthew Waters

08 October 2024

Centricular

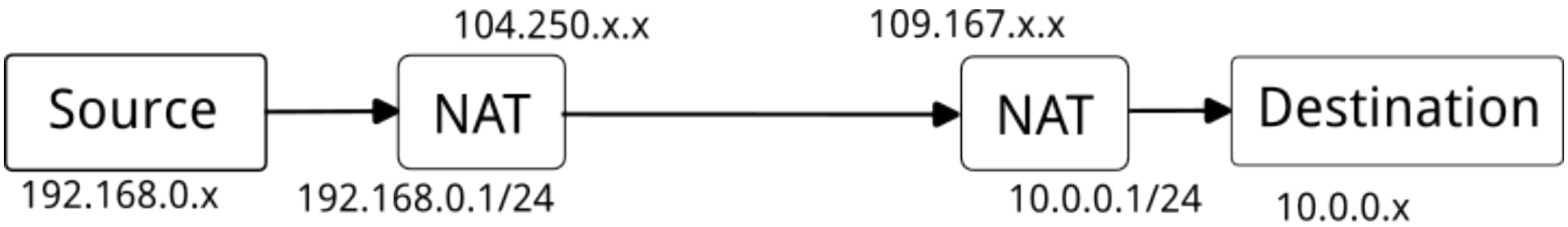
Who Am I?

GStreamer developer and maintainer for over a decade

WebRTC, Vulkan, OpenGL

ICE Overview

See https://gstconf.ubicast.tv/videos/ice-how-to-find-your-way-through-the-internet_22161/



sans-IO

A design pattern where a library does not communicate with external resources

- Allows for increased testability, and reusability of code
- https://sans-io.readthedocs.io/
- https://archive.fosdem.org/2019/schedule/event/rust_sans_io/

ICE (Interactive Connectivity Establishment) Overview

1. Gather candidates
2. Try connecting (Connection Checks)
3. Choose a connection (Nomination)

IO with sans-IO

1. Allocate sockets
2. Pass socket information to sans-IO library
3. sans-IO library notifies of

- What data to send?
- How long to wait for incoming data?
- Whether to close the socket

4. Received data on the socket passed to the sans-IO library

Warning

Prototype code (but functional)

`librice` overview

- librice-proto - sans-IO implementation
- librice - async implementation using `librice-proto`
- `librice-io` - IO implementation
- stun-types - Parsing and writing of STUN messages
- stun-proto - sans-IO implementation of a STUN agent

sans-IO in `librice-proto`

- Available sockets (and STUN servers) are provided via `ComponentMut::gather_candidates`
- Incoming data from a socket is provided by `StreamMut::handle_incoming_data`
- Polling for progress is done using `Agent::poll`

librice-proto features

- UDP and TCP candidates
- STUN gathering
- trickle-ICE
- Exposes a C library interface using `cargo-c` and `cbindgen`

`stun-types`

Parsing and writing `STUN` attributes and messages

- Originally part of `librice` but split out for external use
- `Attributes` are trait based and can be externally defined
- Benchmarks show checking and generating integrity is bottleneck

`stun-proto`

- STUN Agent using sans-IO
- Used by `librice-proto`
- Example stund server (UDP and TCP)

librice-io

Helper IO layer

- Allows sending/receiving on a collection of sockets
- Callback for ready to receive
- Ideally uses the most efficient network API available
  - io_uring, GRO/GSO, etc
- Internally uses the same network and polling API as `async-std` / `smol`
- Very early implementation

Demo

GStreamer WebRTC

Very WIP branch:
https://gitlab.freedesktop.org/ystreet/gstreamer/-/commits/webrtc-rice/

GStreamer integration

- Uses the GStreamer WebRTC ICE abstraction
- Originally used GIO sockets manually
  - Currently uses `librice-io`

librice-proto future plans

- TURN support
- ICE restart
- Consent freshness
- ICE lite
- mdns candidates

librice open questions

- GObject interface?
- Rust crate vs C library confusion
- Do we need to always access `librice-proto` using a C interface?

Thanks

- https://crates.io/crates/librice-proto
- ystreet00 on #gstreamer on OFTC
- https://discourse.gstreamer.org/u/ystreet00
- https://gitlab.freedesktop.org/ystreet
- ystreet00@floss.social on mastodon